Homework 5

1. Fitting non-quadratic losses to data. In `non_quadratic.json`, you will find a $500 \times 300$ matrix $U_{\text{train}}$ and a 500-vector $v_{\text{train}}$ consisting of raw training input and output data, and a $500 \times 300$ matrix $U_{\text{test}}$ and a 500-vector $v_{\text{test}}$ consisting of raw test input and output data, respectively. We will work with input and output embeddings $x = \phi(u) = (1, u)$ and $y = \psi(v) = v$. Our performance metric is the RMS error on the test data set.

In `regression_fit.jl` we have also provided you with a function

$$ \text{regression_fit}(X, Y, l, r, \lambda) $$

This function takes in input/output data $X$ and $Y$, a loss function $l(y_{\text{hat}}, y)$, a local regularizer function $r(\theta)$, and a local regularization hyper-parameter $\lambda$. It outputs the model parameters $\theta$ for the RERM linear predictor. You must include the Flux and LinearAlgebra Julia packages in your code in order to utilize this function. You will use this function to fit a linear predictor to the given data using the loss functions listed below.

- **Quadratic loss:** $\ell(\hat{y}, y) = (\hat{y} - y)^2$.
- **Absolute loss:** $\ell(\hat{y}, y) = |\hat{y} - y|$.
- **Huber loss, with $\alpha \in \{0.5, 1, 2\}$:** $\ell(\hat{y}, y) = p_{\alpha}^{\text{hub}}(\hat{y} - y)$, where

$$ p_{\alpha}^{\text{hub}}(r) = \begin{cases} r^2 & |r| \leq \alpha \\ \alpha(2|r| - \alpha) & |r| > \alpha. \end{cases} $$

- **Log Huber loss, with $\alpha \in \{0.5, 1, 2\}$:** $\ell(\hat{y}, y) = p_{\alpha}^{\text{dh}}(\hat{y} - y)$, where

$$ p_{\alpha}^{\text{dh}}(r) = \begin{cases} r^2 & |r| \leq \alpha \\ \alpha^2(1 - 2\log(\alpha) + \log(r^2)) & |r| > \alpha. \end{cases} $$

We won’t use regularization so you can use $r(\theta) = 0$ and $\lambda = 1$ (though your choice of $\lambda$ does not matter).

Report the training and test RMS errors. Which model performs best? Create a one-sentence conjecture or story about why the particular model was the best one.

**Julia hint.** You will need to define the loss functions above in Julia. You can do this in a compact (but readable) form by defining the function inline, for example, for quadratic loss, $l_{\text{quadratic}}(y_{\text{hat}}, y) = (y_{\text{hat}} - y)^2$. You will also need to do the same for the regularizer (although it is zero); you can do this with $r(\theta) = 0$. 

1
2. **How often does your predictor over-estimate?** In this problem, you will identify how often linear predictors with tilted absolute losses over-estimate.

In `residual_props.json`, you will find a $500 \times 10$ matrix $U_{\text{train}}$ and a 500-vector $v_{\text{train}}$ consisting of raw training input and output data, and a $500 \times 10$ matrix $U_{\text{test}}$ and a 500-vector $v_{\text{test}}$ consisting of raw test input and output data, respectively. We will work with input and output embeddings $x = \phi(u) = (1, u)$ and $y = \psi(v) = v$, and use no regularization ($r(\theta) = 0$). You will also use `regression_fit.jl` from the previous problem.

Recall that the tilted absolute penalty is

$$p_\tau(u) = \begin{cases} -\tau u & u < 0 \\ (1 - \tau)u & u \geq 0, \end{cases}$$

where $\tau \in [0, 1]$. Fit a linear predictor to the given data using the tilted absolute penalty, i.e., $\ell(\hat{y}, y) = p_\tau(\hat{y} - y)$, for $\tau \in \{0.15, 0.5, 0.85\}$. For both the training set and the test set, report how frequently this predictor over-estimates, and plot the empirical CDFs of the residuals.

*Hint.* A predictor $\hat{y}$ over-estimates $y$ when $\hat{y} > y$. To generate an empirical CDF plot of the residuals $r$ of length $d$, you may plot `collect(1:d)/d` versus `sort(r)`.